```
#!/usr/local/bin/perl5

# [1] read the mdxChanges file into an associative array: mdxChanges.
============================
# 1_mkm3dRev "NONE"
# 1_mkmApprovedBy "MIKE MARTYN"
# 1_mkmApprovedDate "12/05/02"

#====Usage====
if (($#ARGV +1) <3) {&usage; exit;}


$mdxChangesFile = $ARGV[0];
open (CHANGES, $mdxChangesFile) or die "can't open";
%mdxChanges = (); #first put empty into the hash

while (<CHANGES>) {
  @line = split / /,$_,2;
  #until ($line[1] =~ /^.*\"$/) {
  #   push @mynewline, $line[1];
  #   print "@mynewline\n";
  #   $line[1] = @mynewline;
  #
  #   #$newline = <CHANGES>; #chomp $newline; #print $newline;
  #   #$line[1] = $line[1] . " " . $newline;
  #   print $line[1];
  #}

  if ($line[1] =~ /^\"(.*)\"$/) {
     $mdxChanges{$line[0]} = $1;
    #$mdxChanges{$line[0]} = $line[1];
  }

}
print "CHANGES NEEDED \(mdxChanges array\):\n";
@keys = keys %mdxChanges;
foreach $key (keys %mdxChanges) {
  print $key, "=", $mdxChanges{$key}, "\n";
}
print "\n\nALL mkm INFOS AND VALUES IN MI FILE:\n";
#================================================================================
==================
# [3] open for read the MI file
$mifile = $ARGV[1];
open (MIFILE, $mifile) or die "can't open original mifile\n";

# [4] open for write the newMI file
$newmifile = $mifile . ".new";
open (NEWMIFILE, ">$newmifile") or die "can't open newmifile\n";

#================================================================================
==================
# Read MI file
# [5] find all ASSP elements, gen an array of element number to element info
value
%assp_array = ();
while ($line = <MIFILE>) {
   if ($line =~ /^ASSP$/) {
      print NEWMIFILE $line;
```

```
      $element_number = <MIFILE>; print NEWMIFILE $element_number; chomp
$element_number;
      $line = <MIFILE>;
      until ($line =~ /^\|\~$/) {
         $element_value = $line;
         print NEWMIFILE $line;
         $line = <MIFILE>;
      }
      print NEWMIFILE $line;
      if ($element_value =~ /_mkm/) {
        $assp_array{$element_number} = $element_value;
      }
      #$assp_array{$element_number} = $element_value;
#================================================================================
==================
# [6] find all TEX elements, gen tex_element array for each TEX element
   }elsif ($line =~ /^TEX$/) {
      $change_needed = 0;
      push @tex_element, $line;
      # Get to the infos list
      for ($i=0;$i<6;$i++) {
         $line = <MIFILE>;
         push @tex_element, $line;
      }
      $count = $line; # The next $count lines are the info numbers that need to
be checked against assp_array
      # Capture the infos list; inspect each info against the assp_array
      for ($i=0;$i<$count;$i++) {
         $line = <MIFILE>;
         push @tex_element, $line;
         #compare $line to assp_arry; if match, set $change_needed to 1, put new
text into $new_text
         #@keys = keys %assp_array;
         chomp $line;
         $info_num = $line;
         $new_text = "";
         #if one of the infos in the element infos matches one of the
assp_infos, a change is needed.
         foreach $key (keys %assp_array) {
            if ($info_num =~ /^$key$/) {
               $newkey = $assp_array{$key}; chomp $newkey;
               $change_needed =1;
               #print "\nkey is: *$key* value is: *$mdxChanges{$newkey}*\n";
               $new_text = $mdxChanges{$newkey};
            }
         }
         #if ($assp_array{$line}) {print $assp_array{$line};}
      }
      $line = <MIFILE>;
      until ($line =~ /^\|\~$/) {
        push @tex_element, $line;
        $line = <MIFILE>;
      }
      push @tex_element, $line;
      #TEX element is completely captured and change_needed is recorded
      if ($change_needed) { #===a change is possibly needed... have to check the
changes array
         # Write the new value to $#tex_array-2
         $info = $assp_array{$info_num};
         chomp $info;
         if ($new_text) {
```

```
            $new_text = "$new_text\n";
            print ("\n", $info, " needs change from *",
$tex_element[$#tex_element-2], "* to: *", $new_text,"*");
            $tex_element[$#tex_element-2]=$new_text;
          }
      }
      print NEWMIFILE @tex_element;
      @tex_element = ();
   }else{
     print NEWMIFILE $line;
   }
}
close MIFILE;
close NEWMIFILE;
unlink $mdxChangesFile;

#@keys = keys %assp_array;
#foreach $key (sort keys %assp_array) {
#  print $key, '=', $assp_array{$key};
#}
#================================================================================
==================


# -read TEX elements
#   -capture each TEX element's array of info_element_numbers
#   -check this info_element_numbers array against the assp_infos_array
#       to lookup the info_text
#   -check the info_text against the changes array.
#       -if there is a match, get the

# new tex value from the changes_array and write it
#        to the TEX element as newMI file is being written.

# ------------------------------------------------------------------------
---------------
#     changes_array        |      assp_infos_array   |  TEX_element |
tex_element_info_array  |
# ------------------------------------------------------------------------
---------------
# 1_mkmRev X1              | 5 1_mkmRev              |  78          | 3
|
# 1_mkm3dRev X2            | 6 HP_BORDER             |  2           | 5
|
# 1_mkmRevDescr AS ISSUED  | 7 DOCU_PARAM#Rev Value  |  0           | 7
|
# ------------------------------------------------------------------------
---------------

# Each TEX element needs to be captured into a tex_element array
# Each info number of the tex_element needs to be captured into a
tex_element_infos array
# Use each tex_info_array element as an index into the assp_infos_array to
lookup the
#    text value of that info
# Check that info text value as an index into the associative changes_array
#   if there is a match (the TEX element needs to be rewritten)
#     take the text value from that element of the changes_array and
#       write it to the TEXT value of the TEX_element_array.
#     Write the adjusted TEX_element_array to the newMI file.
#   else
```

```
#      write the tex_element to the newMI file.

#===SUBROUTINES=====
#===USAGE MESSAGE===
sub usage {
   $0 =~ s%.*/%%;
   print STDERR <<EOUsage;
   Usage: $0 <changes_file> <original MI file> uniqID

EOUsage
   #\033&v3SUsage: $0
}
```